JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Vision-to-Language Intelligence: A Deep Learning Framework for Automatically Generating Image Captions using MobileNetV2 and LSTM

Madhuri Ganesh Dange[1], Shamma Ayubkhan Pathan[2], Krushna Ankushrao Kale[3], Prachi Vishnu Wankhede[4]

[1,2,3,4] Department of Computer Science and Engineering
[1,2,3,4] CSMSS', Chh. Shahu College Of Engineering, Aurangabad (MH) India

Abstract:

AI keeps getting smarter. Now we've got robots that can look at a photo and describe what they see almost like a real person would. For our project, we mixed computer vision with natural language processing so the system can automatically write captions for any image.
Here's the basic idea: MobileNetV2 checks out the picture—it's a fast neural network—and then an LSTM network builds the caption, one word at a time.
We fed the model thousands of photos from the Flickr8k dataset during training, so it's actually solid at figuring out what's going on in a photo. If you want to try it, Streamlit's simple interface lets you upload an image and the model spits out a caption right away.
During testing, it wasn't just tossing out random phrases. It really "got" the pictures and came up with captions that matched what was actually there. This kind of technology makes the whole interaction between humans and machines way smoother, helps monitor online content, supports accessibility for people with disabilities, and pulls together image recognition and language in a pretty practical way.

Keywords:

Artificial Intelligence, Computer Vision, Deep Learning, Image Captioning, MobileNetV2, LSTM, CNN, NLP, Streamlit, and the Vision-Language Model.

I. Introduction

AI's really taken off in recent years, especially in teaching computers not just to look at the world, but to actually talk about what they see. Getting a computer to look at a photo and then lay out what's going on—in plain English—is a pretty wild challenge, but a cool one. It's where computer vision meets how we use language. Put those two together, and suddenly computers aren't just seeing shapes and colors—they're describing sunsets, city streets, or a puppy chasing a ball, just like we might.

Deep learning changed the game here. Early on, image analysis meant people had to design clever tricks by hand, but those ideas could only go so far and didn't scale well. Then came CNNs (Convolutional Neural Networks). These are like supercharged pattern spotters—they learn right from the pixels. On the language

side, RNNs (and especially LSTMs) stepped up, letting computers predict and string together sentences one word at a time. Today, most captioning systems rely on CNNs to "see" and RNNs to "say."

You've probably heard of models like "Show and Tell," "Show, Attend and Tell," or "Neural Image Caption." They proved that if you combine networks that handle visuals and words, you get pretty accurate captions. These systems got surprisingly good at describing not just objects, but actions and scenes. Still, there was a trade-off—all that power means lots of computation and big datasets. That makes things tough if you want real-time results or you're trying to run things on cheaper devices. Train on something small like the Flickr8k dataset, and those same models fumble—captions get awkward or miss little details.

To get around that, lighter models like MobileNetV2 showed up. They're built to hit a sweet spot: fast, without giving up too much accuracy. MobileNetV2 cuts down on heavy processing by using tricks like depthwise separable convolutions and inverted residual blocks (yeah, jargon galore, but basically they're efficiency hacks), yet it still pulls meaningful stuff from images. The system in this paper uses MobileNetV2 to grab image features, then leans on LSTM for turning those features into words. It scales up pretty well and stays quick without making captions meaningless.

Plus, it's wrapped in a simple Streamlit web app. That means anyone can just upload a photo and get an instant caption, right then and there. No need to be a machine learning whiz—just drag, drop, and go. It's perfect for classrooms, social media, healthcare, or any tech space that needs to keep things simple but smart. The point's to show that you can make helpful, real-world captioning both efficient and friendly.

And honestly, these systems could make a real difference. They open up the world for people with vision loss by translating images into words. They save time for people organizing endless photo collections by adding info automatically. As visuals flood the internet, captions make things easier to search and handle. Really, this kind of work helps machines understand and speak about the world a bit like we do, closing the gap between sight and language..

Motivation

So why put all this effort in? Because computers need to get better at actually seeing and describing stuff, not just recognizing it. With images exploding online, writing captions by hand just can't keep up. We need models that can look at photos and instantly come back with clear, everyday English descriptions—something everyone can use. This project focuses on a captioning system that's not just smart and accurate, but also efficient enough to actually use in real time, even if you don't have high-end hardware. It's about making vision AI practical for real life.

Objectives of the Study

1. To learn how to use CNNs to quickly find things in photographs.
2. Learn how to use LSTM networks to make captions in natural language.
3. Learn how to combine language and visual models together to make good caption synthesis.
4. 4. To find out how well the suggested model works, utilize methods like BLEU and accuracy ratings.
5. How to utilize Streamlit to make a web-based tool for producing subtitles in real time.

Scope of the Study

We're working on a deep learning system that creates real-time image captions—ones that actually make sense for each picture. The system grabs image features using MobileNetV2, then uses LSTM to turn those

features into sentences. It's quick, and accuracy stays solid. Right now, we train and test everything with the Flickr8k dataset, but there's room to scale up to something like MSCOCO down the line. This project is just the beginning. It opens doors for better accessibility, easier info management, smarter image searches, and a smoother way for people to interact with computers.

II. Existing System

Thanks to big leaps in computer vision and AI, researchers have been putting a ton of effort into figuring out how to caption photos automatically. The newest approach mixes image recognition with natural language processing, so you get captions that actually make sense. Old systems struggled since they relied on rigid rules and template-based captions, which honestly didn't adapt well to different situations. But the modern methods are way more flexible — they use CNNs to spot visual features and RNNs like LSTMs or GRUs to build out sentences. This mix has shifted the whole field.

Makandar and Suvarnakhandi [21] rolled out a CNN-LSTM setup for captioning that grabs visual features with a VGG16 network and hands them off to an LSTM decoder to produce English sentences. For training and tests, they used the Flickr8k dataset — 8,000 pictures, each with five captions. Their work showed how you can match picture content to natural language if you let a CNN handle the image encoding and an LSTM stitch together the words. It laid the groundwork for future captioning projects and also highlighted the usefulness of BLEU scores for breaking down and evaluating models.

Mehzabeen Kaur and Harpreet Kaur [22] pushed things further by designing a hybrid deep learning model that pulls from VGG16, ResNet50, and YOLO in a multi-encoder framework. This approach captures both objects and context and sends the info to a BiGRU-LSTM decoder. By throwing multiple CNN encoders into the mix, their system managed to grasp more complex details—connections between objects, cues from the environment. The model nailed really high accuracy on Flickr8k, showing how blending features and using transfer learning can quickly boost performance.

Mohammed Inayathulla and Karthikeyan [23] took photo captioning into the video world, focusing on keyframes and tag-generating summaries. They extracted features with DenseNet201 and built a language model using GloVe embeddings and LSTM. The result—videos get clearer and easier to follow since the system turns major visual moments into simple, readable captions. It's genuinely handy for surveillance, browsing, and content discovery.

Iwamura et al. [24] experimented with combining attributes for motion and object detection to make captions richer. Their Motion-CNN incorporated action verbs and temporal hints, blending static and dynamic insights from object regions. When tested on datasets like MSR-VTT2016-Image and MS COCO, the model helped people understand what's happening in the scene—not just what's there, but how things move and interact. Cleaning out background noise made the captions much more informative.

Ahatesham Bhuiyan et al. [25] came up with a new captioning method for Bengali. They used a ResNet-50 encoder and BiGRU decoder with an attention mechanism that shifts focus based on what matters most in the scene. This context-sensitive approach made captions tighter and more relevant, especially in morphologically complex languages like Bengali. On BAN-Cap and BanglaLekhaImageCaption datasets, their model's METEOR scores left older CNN-RNN setups in the dust.

All these methods show how far we've come: combining convolutional feature extraction with sequence modeling is pretty effective. Still, we're not there yet. We need models that can catch tiny details, keep language smooth and connected, and preserve meaning between objects. If we want captions that rival what

humans produce, we'll need even smarter systems, ones can bring in multi-level visual understanding, real contextual emphasis, and better semantic reasoning.

Proposed System

This method uses both computer vision and natural language processing to automatically add captions to pictures, so you don't have to do anything additional.It all revolves around an encoder-decoder model. MobileNetV2 works as the encoder, scanning each photo to pick out what's there. After that, the LSTM decoder grabs those details and turns them into clear, readable captions. The whole system stays light and precise, so it works well even if you're low on computing resources.

A. System Architecture Overview

There are four essential pieces to this setup: preprocessing photos, extracting features, generating sequences, and putting together captions.You upload a photo through a web interface hooked up to MobileNetV2. The system tweaks and resizes your image so it looks right, then the CNN encoder digs out all the spatial and semantic details and hands them off to the decoder. That decoder turns these details into a proper sentence.

The cool part is, you can swap out pieces whenever you want — the framework's modular, so changing the encoder or decoder is pretty easy. That means it's ready for upgrades, like plugging in new multimodal systems or transformer-based decoders

B. Image Feature Extraction Using MobileNetV2

We picked MobileNetV2 as the encoder because it's fast and doesn't chew up a ton of resources. It uses depthwise separable convolutions, so it grabs deep, layered information from images without needing a huge model. To make sure it fits our captioning task, we fine-tune the pretrained MobileNetV2 on the Flickr8k dataset. By dropping its fully connected layer, we turn feature maps into fixed-length vectors. These aren't for classifying images—they're for generating captions.

This approach makes sure the system catches vital visual info: objects, their features, and spatial layouts. The decoder takes all of that and builds sentences that actually sound natural.

C. Caption Generation Using LSTM

The decoder, an LSTM network, takes the encoder's visual feature vector and starts processing word embeddings, one by one, to predict the next word in your caption. LSTM networks are great at holding onto long-term dependencies, so the generated sentence keeps its meaning and structure

The system uses an embedding layer, like GloVe or Word2Vec, to turn words into dense vectors that show how words relate to each other. With the encoded image and The decoder learns to guess what comes next from the words before, so the captions make sense and stick to what's happening in the image.

D. Dataset and Preprocessing

We use the full Flickr8k collection: 8,000 pictures, each with five distinct captions. Before training, the captions are split up into tokens, changed, and made the same length. To make things easier for the model, rare terms are thrown out. Images are scaled to 224x224 pixels and made normal for MobileNetV2.

We also augment the data with flips, rotations, and color changes to keep it varied and reduce too much fitting. The dataset is divided into three parts: training, validation, and testing. The ratio is 80:10:10.

E. Model Training and Optimization

The encoder and decoder both learn from scratch using the training set. During training, the decoder learns to figure out what the captions are by using the words that have been created so far and the features that the encoder has retrieved. The Adam optimizer helps us identify the ideal learning rate, and for classification, the loss function is cross-entropy.

Teacher forcing helps the model make better predictions, speeds up training, and leads to more fluent sentences. BLEU scores measure how well the model's captions match up with the reference ones.

F. Web Interface Implementation

We used Streamlit to make the web app, which is easy to use and pretty enjoyable. You upload a picture, and the UI makes a caption right away. The trained model loads the image, processes it, pulls out features, and writes out the description. The caption pops up at the end

This approach makes the technology easy to use in education, research, or for helping people. For example, it's a real boost for visually impaired users — the computer does the heavy lifting, describing what's in their photos.

G. Advantages of the Proposed Model

There are a lot of good things about the suggested MobileNetV2–LSTM framework:

1. **Lightweight and Fast:** MobileNetV2 makes sure that less computational power is needed, which makes it great for use on edge devices in real time.
2. **High Accuracy:** The LSTM decoder does a fantastic job at understanding how words relate to one other, which makes the captions sound better.
3. **Generalizability:** The model works well with many different kinds of images.
4. **Ease of Integration:** The modular design lets you connect to more complex NLP transformers in the future.
5. **User-Friendly Interface:** The Streamlit platform makes it easy to use and exhibit in real life.

III. System Design

This photo captioning model keeps things simple. It's built in layers and modules, so you can see exactly how everything fits together. If you want to make changes or add new features, that's not a headache. The system employs computer vision and natural language processing to make its own descriptions of pictures.Different parts of the setup handle different jobs—like grabbing data, writing captions, that sort of thing.

Here's the basic process in five steps:
(1) Load and prep your photos.
(2) Use MobileNetV2 to pull out features.
(3) Rely on LSTM to generate the captions.
(4) Train the model, then test it.
(5) Wrap it all up in a Streamlit web app.
This architecture makes sure that everything is done automatically, from observing something to writing about it.

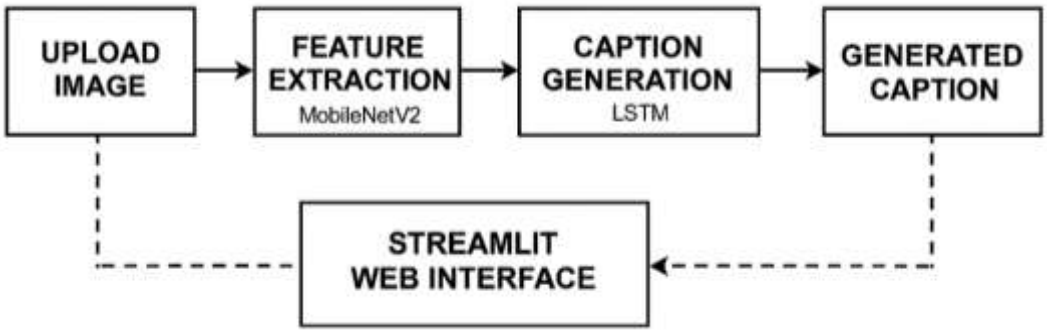

Fig. 1 System Architecture

A. System Architecture

This setup sticks to the classic encoder-decoder style. MobileNetV2 works as the encoder, pulling out all the essential info from any photo you upload. Then the LSTM decoder jumps in and turns those details into a simple, clear sentence describing what's going on in your picture.

1. **Input Layer:** You upload a photo through the web app.
2. **Preprocessing Layer:** he system resizes it to 224 by 224 pixels, normalizes it, and turns it into a tensor so MobileNetV2 can process it.
3. **Encoder Layer:** MobileNetV2 scans the image and grabs a deep, high-dimensional vector—it's like a summary of who's in the scene, what they're doing, and the overall context.
4. **Decoder Layer:** fter that, the LSTM decoder takes over. It reads that vector and writes a caption, picking just the right words to describe what's happening.
5. **Output Layer:** he Streamlit app then shows you this caption.

The encoder and decoder really work together, bouncing info between each other so every photo gets turned into meaningful text. And honestly, this design is pretty flexible you can tweak things as new tech comes out or swap out the models whenever you feel like upgrading.

B. Data Flow Design

Here's how it works:

1. **Image Acquisition:**

First, you upload your photo in the app. It's simple—just choose your image and that's it.

2. **Image Preprocessing:**

Once your photo hits the app, the system gets started. It switches up the format, resizes everything, and makes sure it's all set for what comes next.

3. **Feature Extraction:**

MobileNetV2 jumps in at this point. It scans your photo and pulls out a feature vector, so all the important parts—like objects and scenes—get bundled up into a quick summary.

4. **Tokenization and Embedding:**

After that, the system grabs the training captions, breaks them down into individual words (tokens), turns those words into numbers, and squashes everything into dense vectors. This step helps the system understand how the words connect.

5. **Caption Generation:**

Now comes the decoder. It uses LSTM to dive into your image info and builds your caption, word by word. It decides what word fits next, lays it out step by step, and finishes when the caption's ready.

6. **Result Display:**

Streamlit pops the caption right onto your screen. Your photo is turned into text in just a few seconds. Honestly, it's pretty cool.

C. System Modules

1. **Image Preprocessing Module:**

his module changes the size of photos, makes them all the same size, and adds to them. It makes sure that all the pictures meet the MobileNetV2 criteria and improves the model.

2. **Feature Extraction Module:**

Uses MobileNetV2 as the encoder to get important features from photos.We adjust the parameters of the pretrained network on the Flickr8k dataset so that they suit patterns that are only seen in that domain.

3. **Language Modeling Module:**

Uses LSTM to show how captions are put together. It uses the encoded visual hints to guess what the next word will be in a series.

4. **Training & Optimization Module:**

It trains both the encoder and decoder at the same time using the right loss functions (categorical cross-entropy) and optimizers (Adam). People work together to save time on math.

5. **Evaluation Module:**

Checks the quality and fluency of the captions that were made by using performance metrics like BLEU score and accuracy.

6. **User Interface Module:**

This module makes an app with Streamlit that lets people share photos and obtain captions right away. It gives you a fun, straightforward, and easy-to-use space to test out the system.

D. Working Steps of the Proposed System

1. **Step 1:** Use the Streamlit online interface to upload an image.
2. **Step 2:** Prepare the picture to change size and normalising.
3. **Step 3:** Use MobileNetV2 to get deep visual features.
4. **Step 4:** Send the LSTM-based decoder the encoded features.
5. **Step 5:** Make caption words one after the other till you reach an end token.
6. **Step 6:** Show the user the caption that was made in real time.

This sequential workflow makes sure that each image is processed in a precise order so that the captions are correct and make sense.

E. Design Considerations

The three fundamental ideas that guide the design are efficiency, accuracy, and scalability. MobileNetV2 doesn't need as much computing power, thus it can be used and deployed in real time on edge devices. We picked LSTM because it has a strong history of being able to capture language model dependencies across lengthy periods of time. We utilize Streamlit, a lightweight web framework, to connect everything. This makes it easier to set up and use.

By replacing outdated parts with new ones, such Vision Transformers (ViT) or transformer-based decoders like BERT or GPT, you can make things work better in the future. The way the system was made makes this possible.

IV. Expected Outcome

The proposed Image Caption Generator must generate live captions that are grammatically accurate, contextually relevant, and compatible with a diverse array of photographs. The goal is to design captions that computers can easily read and that sound like how people truly communicate. It does this by using MobileNetV2 to find sections of pictures and LSTM to place them in order. The end goal is to construct a

model that can deal with complicated visual circumstances, find a lot of items, and give descriptions of pictures that make sense and illustrate what they are and how they are related.

The system should perform well on normal computer hardware and score higher on BLEU tests than current CNN-LSTM models. MobileNetV2 speeds up inference and puts less strain on computers, all while making the captions simpler to read. The model should do well with a lot of different kinds of photos, like landscapes, objects, animals, and humans, after being trained on the Flickr8k dataset. You should be able to quickly submit in pictures and obtain back descriptions using the Streamlit online interface. It should also be fun and easy to use. The system can handle data in real time, which means it can be used in real life to automate social media, find pictures, arrange pictures, and make things easier for people who can't see.

This research demonstrates that the application of computer vision and natural language processing can assist in addressing challenges encountered throughout the learning process in multiple ways. The anticipated result facilitates the convergence of human and machine interpretation by augmenting intelligent systems' ability to comprehend and convey visual information. Better models that leverage attention-based vision-language models or more advanced architectures like Transformers will build on the current paradigm.

V. Conclusion

The Image Caption Generator leverages MobileNetV2 and LSTM to combine computer vision and natural language processing to automatically figure out what captions mean for digital photos. The method illustrates that deep learning can bridge the gap between spoken and written language by turning pictures into standard English. The model works well in real time because it is fast and has nice subtitles. This is possible because it has a lightweight encoder and a decoder that works in the appropriate order. Streamlit makes it easy to employ AI-powered visual interpretation systems by letting you create a web interface for them. It also shows how they work. This study shows that convolutional neural networks (CNNs) and recurrent neural networks (RNNs) can work together to make smart captions for a wide range of image datasets that are very similar to English written by people.

VI. Future Scope

This effort attempts to enhance the precision of captions and their contextual relevance by employing advanced architectures in the future, such as Vision Transformers (ViT) and Transformer-based language decoders (BERT, GPT, or T5). To get better at generalizing across a wider range of image domains, it is best to train with bigger datasets like MS COCO or Flickr30k. If you add attention processes and semantic segmentation to the model, it might be able to explain things better. This would help it focus more on the most important parts of photographs. Using optimized frameworks like TensorFlow Lite on mobile and edge devices can also make it easier to use the model in real time in assistive technologies, surveillance systems, and multimedia content management. The idea might be broadened to incorporate subtitles in multiple languages, so that people all over the world who speak different languages can understand them.

References

Bonk, C. J., Lee, M. M., Reeves, T. C., & Reynolds, T. H. (Eds.). (2015). MOOCs and K Masters. A Brief Guide To Understanding MOOCs. The Internet Journal of Medical Education. 2011 Volume 1 Number 2.

K Masters. A Brief Guide To Understanding MOOCs. The Internet Journal of Medical Education. 2011 Volume 1 Number 2.

[1] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” arXiv preprint arXiv:1409.1556, 2015.
[2] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 770–778.
[3] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and Tell: A Neural Image Caption Generator,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2015, pp. 3156–3164.
[4] J. Donahue, L. Hendricks, S. Guadarrama, and M. Rohrbach, “Long-Term Recurrent Convolutional Networks for Visual Recognition and Description,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2015, pp. 2625–2634.
[5] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.
[6] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” arXiv preprint arXiv:1301.3781, 2013.
[7] K. Cho et al., “Learning Phrase Representations Using RNN Encoder–Decoder for Statistical Machine Translation,” arXiv preprint arXiv:1406.1078, 2014.
[8] C. Szegedy et al., “Going Deeper with Convolutions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2015, pp. 1–9.
[9] Y. LeCun, Y. Bengio, and G. Hinton, “Deep Learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015.
[10] A. Karpathy and L. Fei-Fei, “Deep Visual–Semantic Alignments for Generating Image Descriptions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2015, pp. 3128–3137.
[11] J. Johnson, A. Karpathy, and L. Fei-Fei, “DenseCap: Fully Convolutional Localization Networks for Dense Captioning,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 4565–4574.
[12] P. Anderson et al., “Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2018, pp. 6077–6086.
[13] R. Krishna, Y. Zhu, O. Groth et al., “Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations,” Int. J. Comput. Vis., vol. 123, pp. 32–73, 2017.
[14] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel, “Self-Critical Sequence Training for Image Captioning,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2017, pp. 7008–7024.
[15] A. Vaswani et al., “Attention Is All You Need,” in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2017, pp. 5998–6008.
[16] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, “Show, Control and Tell: A Framework for Generating Controllable and Grounded Captions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2019, pp. 8307–8316.
[17] J. Lu et al., “ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks,” Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 32, 2019.
[18] X. Chen et al., “Microsoft COCO Captions: Data Collection and Evaluation Server,” arXiv preprint arXiv:1504.00325, 2015.
[19] T. Yao, Y. Pan, Y. Li, and T. Mei, “Exploring Visual Relationship for Image Captioning,” in Proc. Eur. Conf. Comput. Vis. (ECCV), 2018, pp. 684–699.
[20] A. Fang, X. Lin, D. Yang, and S. Wang, “Semantic-Guided Transformer for Vision–Language Understanding,” IEEE Trans. Multimedia, vol. 25, pp. 12–24, 2023.
[21] Dr. Aziz Makandar and Keerti Suvarnakhandi, “Image Caption Generator Using CNN–LSTM,” International Journal of Advances in Engineering and Management (IJAEM), vol. 4, no. 9, pp. 122–129, 2022.

[22] Mehzabeen Kaur and Harpreet Kaur, "An Efficient Deep Learning-Based Hybrid Model for Image Caption Generation," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 14, no. 2, pp. 101–110, 2023.

[23] Mohammed Inayathulla and Karthikeyan C, "Image Caption Generation Using Deep Learning for Video Summarization Applications," IJACSA, vol. 15, no. 4, pp. 215–222, 2024.

[24] Kiyohiko Iwamura, Jun Younes Louhi Kasahara, Alessandro Moro, Atsushi Yamashita, and Hajime Asama, "Image Captioning Using Motion-CNN with Object Detection," Sensors, vol. 21, no. 12, p. 1270, 2021.

[25] Ahatesham Bhuiyan, Eftekhar Hossain, Mohammed Moshiul Hoque, and M. Ali Akber Dewan, "Enhancing Image Caption Generation Through Context-Aware Attention Mechanism," Heliyon, vol. 10, no. 2, p. e15231, 2024.